SE Experiment-3
(Batch-A/A1)

-Team Details-
Mahadev Balla – UID (2023300010)
Daksh Bari – UID (2023300012)
T.E. Computer Engineering – A

Aim:

To draw Class diagram for a Melanoma Detection Web Application.

Implementation:

The development of the class diagram followed a systematic methodology. First, the problem statement was thoroughly analyzed to extract key nouns and noun phrases, which were subsequently refined to eliminate redundancy. These refined terms were then categorized into potential classes. The interactions and relationships between these classes were carefully mapped to establish the structural foundation of the system. The final class diagram was constructed based on these identified relationships and functionalities.

Problem Description:

The melanoma detection platform is a web-based application designed to facilitate early skin cancer screening. The system supports multiple user roles with distinct capabilities. Unregistered visitors can explore the application's features and create accounts. Authenticated users can submit skin lesion images for AI-powered analysis, receive instant risk assessments, schedule dermatologist consultations, and maintain their medical history. Administrative personnel oversee user management and maintain healthcare professional databases. The platform integrates machine learning for preliminary melanoma risk evaluation and provides seamless appointment management functionality.

Following is a breakdown of various parts that were considered for designing the class diagram-

[1] Noun/Noun Phrases.

User, Patient, Admin, Dermatologist, Image, Prediction, History, Risk Assessment, Appointment, Notification, AI Model, Analysis, Upload, Booking, Cancellation.

[2] Classes.

User, Image Analysis, Dermatologist, SkinImage, PredictionResult, Appointment, NotificationService, AIModelService.

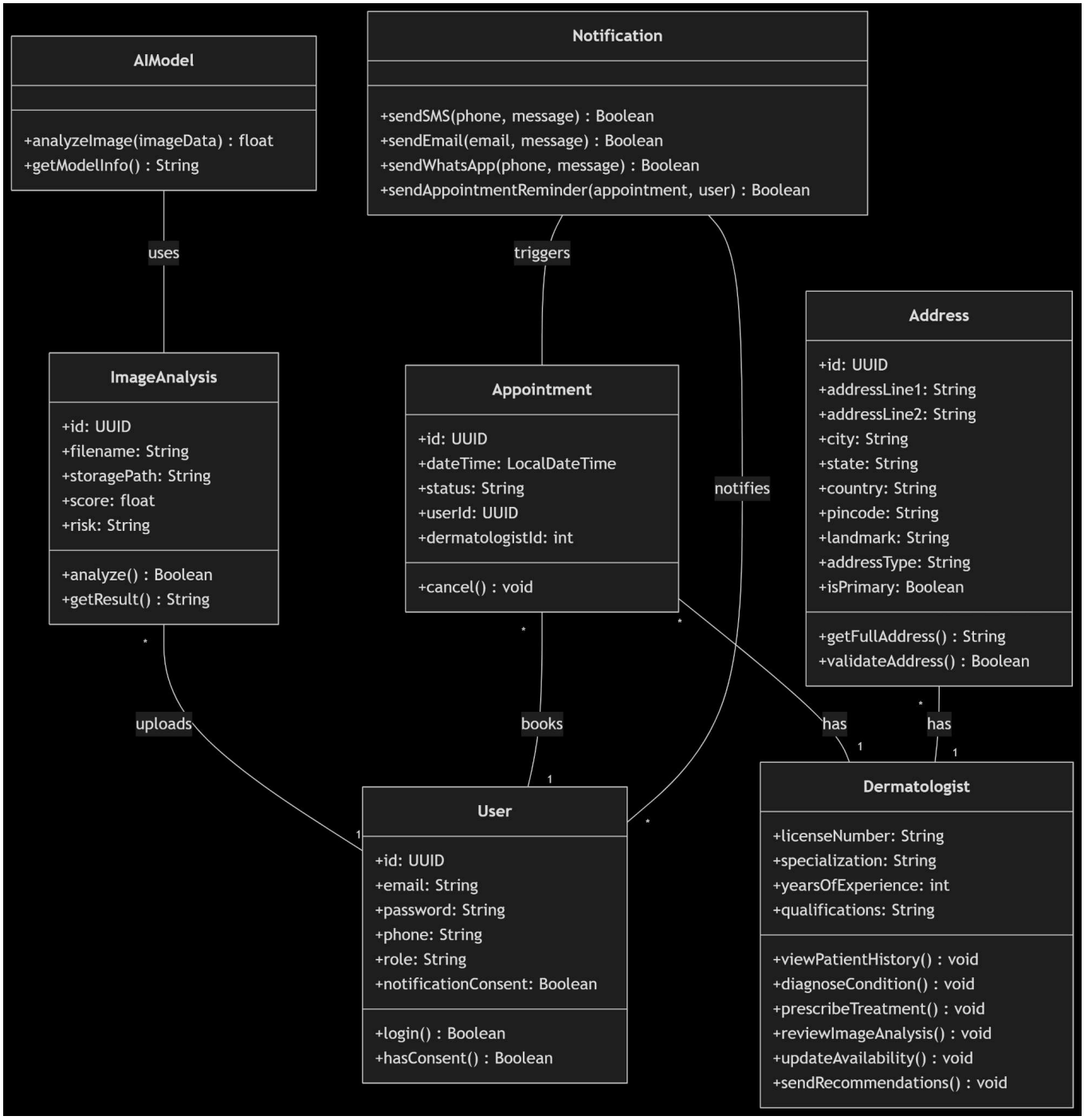
[3] Verb Phrases.

1. Guest User registers on the application.
2. User logs into the system.
3. Patient uploads Skin Image.
4. System analyzes image using AI Model.
5. Patient views Prediction Results.
6. Patient books Appointment with Dermatologist.
7. Patient cancels Appointment.
8. System sends Notification for appointments.
9. Admin manages User accounts.
10. Admin manages Dermatologist records.

[4] Relations.

1. Patient and Admin are different users of the software and hence are generalized to User.
2. SkinImage generates PredictionResult through analysis.
3. Patient creates Appointment with Dermatologist.
4. Appointment triggers Notification.
5. Patient has multiple PredictionResults in history.
6. Patient can have multiple Appointments.
7. Dermatologist can have multiple Appointments.
8. AIModelService analyzes SkinImage.
9. NotificationService notifies User about appointments.
10. Patient must be registered to upload images and book appointments.

[5] Class Diagram.



Following are the code snippets that were written corresponding to each class and interface demonstrated in the above diagram-

User.java

```
public class User {
    private UUID id;
    private String email;
    private String password;
    private String phone;
    private String role;
    private boolean notificationConsent;

    public void login() {}
    public void hasConsent() {}
}
```

Dermatologist.java

```
public class Dermatologist {
    private String licenseNumber;
    private String specialization;
    private int yearsOfExperience;
    private String qualifications;

    public void viewPatientHistory() {}
    public void diagnoseCondition() {}
    public void prescribeTreatment() {}
    public void reviewImageAnalysis() {}
    public void updateAvailability() {}
    public void sendRecommendations() {}
}
```

Appointment.java

```java
public class Appointment {
    private UUID id;
    private LocalDateTime dateTime;
    private String status;
    private UUID userId;
    private int dermatologistId;

    public void cancel() {}
}
```

ImageAnalysis.java

```java
public class ImageAnalysis {
    private UUID id;
    private String filename;
    private String storagePath;
    private float score;
    private String risk;

    public void analyze() {}
    public void getResult() {}
}
```

Notification.java

```java
public class Notification {
    public void sendSMS(String phone, String message) {}
    public void sendEmail(String email, String message) {}
    public void sendWhatsApp(String phone, String message) {}
    public void sendAppointmentReminder(Appointment appointment, User user) {}
}
```

AIModel.java

```
public class AIModel {
    public void analyzeImage(byte[] imageData) {}
    public void getModelInfo() {}
}
```

Address.java

```
public class Address {
    private UUID id;
    private String addressLine1;
    private String addressLine2;
    private String city;
    private String state;
    private String country;
    private String pincode;
    private String landmark;
    private String addressType;
    private boolean isPrimary;

    public void getFullAddress() {}
    public void validateAddress() {}
}
```

Conclusion:

This project helped us learn how to create class diagrams by working on a real example a skin cancer detection app. We successfully designed a clear plan showing all the main parts of the system (like users, appointments, and image analysis) and how they connect to each other. We also wrote the Java code that matches the diagram.Through this, we improved our skills in organizing software, understanding how different pieces fit together, and turning ideas into a structured plan. It was a useful practice that combined learning with hands-on experience.